ZOR – Methods and Models of Operations Research (1993) 38:113–129

Cooperation in One Machine Scheduling

IMMA CURIEL

Department of Mathematics, University of Maryland, Baltimore Country, Catonsville, MD 21228, USA

JOS POTTERS

Department of Mathematics, University of Nijmegen, Toernooiveld, 6525 ED Nijmegen, The Netherlands

RAJENDRA PRASAD

Indian Statistical Institute, 8th Mile, R.V. College Post, Mysore Road, Bangalore 560040, India

STEF TIJS

Faculty of Economics, University of Brabant, P.O. Box 90 15 3, 5000 LE Tilburg, The Netherlands

BART VELTMAN

Center for Mathematics and Computer Science, P.O. Box 40 79, 1009 AB Amsterdam, The Netherlands

Abstract: A combinatorial optimization problem can often be understood as the problem to minimize cost in a complex situation. If more than one party is involved, the solution of the optimization problem is not the end of the story. In addition it has to be decided how the minimal total cost has to be distributed among the parties involved. In this paper cost allocation problems will be considered arising from one-machine scheduling under additive and weakly increasing cost functions. The approach of the problem will be game theoretical and we shall in fact show that in many cases the games related to the cost allocation problems are balanced.

Key Words: cost allocation, cooperative game, one-machine scheduling.

In combinatorial optimization the problem is often to compound – as cheaply as possible – a composite whole out of available components. The minimum cost spanning tree problem, for example, looks for a spanning tree of minimal cost composed from the arcs available in a graph. The traveling salesman problem asks for a hamiltonian cycle of minimal length consisting of arcs of a given graph. When the optimization problem has been solved we are often faced with a second problem, especially if the composition of the whole is in the interest of more than one agent. If, for example, the minimal cost spanning tree problem models the connection of villages with a water resource, then it is important for

0340-9422/93/38:2/113-129 $2.50

the villages to be connected and therefore it seems reasonable that they contribute in the cost. Also in the traveling salesman problem it may be in the interest of the cities (or inhabitants of the cities) to be visited. In those situations the optimization problem should be followed by a cost allocation problem. In the literature there can be found several papers in the same spirit as this one, such as the paper by Granot and Huberman (1981) about the cost allocation in minimal cost spanning tree problems, the paper of Kalai and Zemel (1982) about flow problems and the paper of Potters et al. (1990) about the traveling salesman problem. The paper of Tijs/Driessen (1986) gives a general description of the connection between cost allocation and the theory of cooperative games.

In this paper we elaborate on this general philosophy in the case of one-machine scheduling problems. We assume that associated with each job there is an agent (player) only interested in a quick processing of this particular job. In technical sense this implies that we will consider regular and additive optimality criteria. Further we formulate the optimization problem slightly differently. We assume that the jobs are already given in an initial order and ask for a rearrangement of the jobs with maximal cost savings. In order to get a reasonable distribution of the cost savings among the jobs (agents) we take into consideration the "virtual" cost savings which a subgroup (coalition) of agents can obtain by – for this group admissible – rearrangements. Which rearrangements are admissible for a coalition depends on the special situation, but in general the following two principles should be maintained:

1) The rearrangement should not hurt the interests of the agents outside the coalition.
2) The rearrangement is possible without an active cooperation of agents outside the coalition.

By these rules we sometimes take as admissible rearrangements every permutation of the positions of the jobs in the coalition, at other time we impose stronger conditions (e.g. only permutations which respect the components of the coalition are admissible; see section 1 for a more explicit description).

We finish this introduction with a concrete example.

Suppose an aircraft industry has a waiting list of customers (airlines companies) each waiting for the delivery of one aircraft. Each airline company has to incur cost, weakly increasing with the data of delivery. If the airlines companies cooperate, they can decide to make a new list with minimal cost and to compensate the companies which get their aircraft later delivered for the increase of cost. In this paper we study the problems involved in reaching a suitable compensation.

The organization of the paper is as follows. In section 1 we introduce the concepts from job scheduling and cooperative game theory that we will need. In the sections 2 and 3 we consider special job scheduling situations and prove that the associated cost saving games are balanced. In section 2 the main restriction will be that all jobs have the same processing times. In section 3 we introduce

two generalizations of the sequencing games of Curiel et al. (1989): σ_0-components additive games and σ_0-pairing games. We prove the balancedness of these types of games and a rule to obtain a core element is given. In the last section (section 4) we give some suggestions for future research and a few results already obtained in Hamers (1988) and Veltman (1988).

1 Preliminaries

In this section we introduce the concepts from job scheduling and cooperative game theory needed in the following sections. Further, we describe how we think to combine these theories for the purpose we have in mind.

I) Job Scheduling

We consider single machine scheduling problems with a regular and additive criterion R. The set of jobs will be denoted by $N = \{1, 2, \ldots, n\}$. The jobs are given in an initial order. The position of each job with respect to a certain order can be described by a permutation σ of N. So, $\sigma(i) = j$ means that job i has the j-th position in the order according to σ. By renumbering the jobs we may assume that, initially, the i-th job is on the i-th place. Each job $i \in N$ has a ready time r_i. The processing of job i cannot start before r_i. We assume that the jobs are given with an initial order that causes the ready times to be weakly increasing. So, if the initial order is given by the identity permutation this means that $0 = r_1 \leq r_2 \leq \cdots \leq r_n$. For each job i the *processing time* is denoted by $p_i > 0$ and (if it matters) the due date is $d_i \geq r_i + p_i$. In general, we consider additive regular optimality criteria i.e. for each agent (job) i there is a weakly increasing function f_i: $[r_i + p_i, -) \to \mathbf{R}$ such that the expression $\sum_{i \in N} f_i(C_i)$ is to be minimized. Here the number C_i is the completion time of job J_i. Special examples are the functions $f_i(t) = \alpha_i t$ with $\alpha_i > 0$ for each $i \in N$ (weighted completion time criterion) and the functions $f_i(t) = 0$ for $r_i + p_i \leq t \leq d_i$ and $f_i(t) = a_i$ if $t > d_i$ (penalties $a_i > 0$ for tardiness).

The jobs can be reordered to decrease the total cost. As mentioned above, such a new order can be described by a permutation of N. A *time table* τ is a mapping which assigns to every job i a starting time t_i with $t_i \geq r_i$ for all jobs $i \in N$ and $t_j \geq t_i + p_i$ if $t_j \geq t_i$ and $i \neq j$. As far as we are considering weakly increasing cost functions f_i we may restrict our attention to semi-active time tables, i.e. time tables without unnecessary delay. If we put the jobs in the order σ: $N \to \{1, 2, \ldots, n\}$ (σ assigns ranking numbers to jobs) the starting time of job i in the

semi-active time table determined by σ is

$$t_{i,\sigma} := \max(r_i, t_{j,\sigma} + p_j)$$

where job j is such that $\sigma(j) = \sigma(i) - 1$. If $\sigma(i) = 1$, then $t_{i,\sigma} = r_i$. If σ fixes an order the completion time of i equals $C_i(\sigma) = t_{i,\sigma} + p_i$.

II) Cooperative Game Theory

A *cooperative game* (the terms cooperative game with side payments, game in characteristic function form, game in coalitional form and games with transferable utility are also used) is a pair (N, v) where N is a finite set (of *players*) and v is a mapping $v: 2^N \to \mathbf{R}$ with $v(\emptyset) = 0$. The mapping v assigns to each *coalition* $S \subset N$ the worth of the coalition $v(S)$. A cooperative game (N, v) is called *superadditive* if for all coalitions $S, T \subset N$

$$v(S) + v(T) \leq v(S \cup T) \qquad \text{whenever } S \cap T = \emptyset \ .$$

A cooperative game (N, v) is called *convex* if for all coalitions $S, T \subset N$

$$v(S) + v(T) \leq v(S \cup T) + v(S \cap T) \ .$$

In other areas, e.g. matroid theory, a set function that satisfies the above property is called *supermodular*.

Cooperative games are often used to describe economic situations in which several individuals can benefit by cooperating because the profit of a group consisting of several individuals is greater than the sum of the profit of the separate individuals. If the grand coalition N is formed the problem is how to divide the total profit $v(N)$ among the players. An *allocation* of the amount $v(N)$ can be described by a vector $x \in \mathbf{R}^N$ with $x(N) := \sum_{i \in N} x_i = v(N)$. The quantity x_i is the amount allocated to player i. Such an allocation is called *individual rational* if $x_i \geq v(i)$. If the players decide to divide $v(N)$ according to an individual rational allocation, no player has an incentive to leave the grand coalition N and work on his own because he is receiving at least as much as he would be able to obtain when working alone. However, it could still be possible for a group of players to leave the grand coalition and to do better working as a separate group. This would cause the grand coalition to be unstable, even in situations where the greatest total profit can be obtained if it is formed. If an allocation x satisfies the condition $\sum_{i \in S} x_i \geq v(S)$ for all $S \subset N$, then this problem does not arise because

clearly, no coalition can do better by leaving the grand coalition and working on its own. The set of all allocations that satisfy this condition is called the *core* of the game (N, v) and is denoted by $C(v)$. So, formally we have

$$C(v) := \left\{ x \in R^n \,\middle|\, \sum_{i \in N} x_i = v(N), \sum_{i \in S} x_i \geq v(S) \text{ for all } S \subset N \right\} .$$

The core of a cooperative game can be empty. If it is not empty it is a convex and compact subset of R^n. In general it will contain more than one element if it is not empty. The core of a convex game is always non-empty. A game with a non-empty core is called *balanced*. Given a balanced game (N, v) we can construct a subgame of (N, v) by restricting v to a subset S of N. If all the subgames that can be constructed in this way are also balanced, we call the game *totally balanced*. Two other solution concepts widely used in cooperative game theory are the Shapley value (Shapley (1953)) and the τ-value (Tijs (1981)). Contrary to the core these are one-point solution concepts, that is instead of assigning a set of allocations to each game they assign one particular allocation.

To explain how the Shapley value works we need the following definitions. Let Π_N denote the set of all permutations of the set N. Consider a $\pi \in \Pi_N$. The set $P(\pi, i)$ is defined to be the set of predecessors of i with respect to the permutation π. So $P(\pi, i) := \{ j \in N | \pi(j) < \pi(i) \}$. The vector $\psi^\pi(v) \in R^n$ is defined to be the vector with i-th coordinate equal to $v(P(\pi, i) \cup \{i\}) - v(P(\pi, i))$. The Shapley value $\phi(v)$ is given by

$$\phi_i(v) := n!^{-1} \sum_{\pi \in \Pi_N} \psi_i^\pi(v) \qquad \text{for all } i \in N .$$

The idea behind the Shapley value is the assumption that the formation of the coalition N takes place according to a permutation π, with players joining one after the other in the order $\pi^{-1}(1), \pi^{-1}(2), \ldots, \pi^{-1}(n)$. Player i receives his marginal contribution which is equal to the amount $\psi_i^\pi(v)$. The Shapley value assigns to player i has expected payoff in the case where all possible orders of formation are considered equally likely. The Shapley value of a convex game is the barycenter of the core.

The τ-value is defined on the set of *quasi-balanced games*, a subset of the set of all cooperative games. The set of balanced games is a subset of the set of quasi-balanced games and therefore if a game has a core allocation, then the τ-value can be defined. If (N, v) is a cooperative game, we define the *upper vector* of the game to be the vector $M(v) \in \mathbf{R}^n$ with coordinates $M_i(v) = v(N) - v(N \backslash i)$. $M_i(v)$ is the maximal payoff player i can expect to obtain; if he asks for more the others will be better of without him.

Further, we define the *lower vector* $m(v) \in \mathbf{R}^n$ by $m_i(v) = \max_{S: i \in S} [v(S) - \sum_{j \in S, j \neq i} M_j(v)]$. The game (N, v) is quasi-balanced if $m(v) \leq M(v)$ and

$\sum_{i \in N} m_i(v) \leq v(N) \leq \sum_{i \in N} M_i(v)$. The τ-value of the game (N, v) is the unique vector $\tau(v)$ on the segment $[m(v), M(v)]$ with $\sum_{i \in N} \tau_i(v) = v(N)$. In the following section we only need the fact that for a convex game $m_i(v) = v(i)$ for all $i \in N$.

III) *Sequencing and Cooperation*

If we have a single machine scheduling problem and an initial order of the jobs σ_0, then we can define a cooperative game (N, v) by taking N to be the set of agents behind each job. The characteristic function v of the game can be defined by considering the maximum cost savings that a coalition can achieve. This leads to the following definition for $v(S)$

$$v(S) := \sum_{i \in S} f_i(C_i(\sigma_0)) - \min_{\tau \in A(S)} \sum_{i \in S} f_i(C_i(\tau))$$

where $A(S)$ is the set of time tables, admissible for coalition S. It is clear that $A(S)$ depends on the reorderings that we allow coalition S to perform. In this paper we mostly deal with two cases. The first one is the case where coalition S is allowed to reorder the jobs belonging to S in any way it wants while the jobs not belonging to S do not change positions. The set of permutations that satisfy this property will be denoted by $\Pi_S^{\sigma_0}$. When σ_0 is the identity permutation the notation will be simplified to Π_S. The second case is more restrictive, coalition S is allowed to perform only those reorderings that respect its components according to the original order σ_0. This means that two jobs in S are not allowed to switch their position if there is a job in between, not belonging to S. We will denote the set of permutations satisfying this property by $PC_S^{\sigma_0}$. When σ_0 is the identity permutation this notation will be simplified to PC_S. The idea of components can be formalized in the following way. The *components of a coalition S with respect to the order σ_0* are the equivalence classes in S under the relation $i \sim j$ where $i \sim j$ means that the coalition S contains all jobs k with ranking number $\sigma_0(k)$ between that of i and that of j. For example if the initial order σ_0 is the order $1 \prec 2 \prec 3 \prec 4 \prec 5$ and S is the coalition $\{1, 2, 4, 5\}$, then S consists of two components $\{1, 2\}$ and $\{4, 5\}$ and the switch $(1, 4)$ is *not* allowed in the second situation (they are not allowed to jump over job 3). We denote the set of components of a coalition S with respect to σ_0 by S/σ_0.

Two papers on cost allocation play a crucial role in our discussion.

(1) The first one is the paper of Tijs et al. (1984) where the following situation is discussed. All jobs have processing time *one* and all reorderings of its jobs are allowed to coalition S. The original order of the jobs is taken to be given by the identity permutation. Let A be an $N \times N$-matrix where A_{ij} denotes the cost player i incurs if he takes the j-th position. A permutation game is defined

by

$$v(S) := \sum_{i \in S} A_{ii} - \min_{\sigma \in \Pi_S} \sum_{i \in S} A_{i\sigma(i)} \ .$$

The main result of this paper is that permutation games are totally balanced.

(2) In the paper on sequencing games of Curiel et al. (1989) the situation is different. There the jobs may have different processing times and the weighted completion time criterion $\sum \alpha_i C_i$ is studied. Also, the original order need not be given by the identity permutation. For a coalition S only the permutations which respect the components of S in the original order are allowed. Hence, a sequencing game is defined by

$$v(S) := \sum_{i \in S} \alpha_i C_i(\sigma_0) - \min_{\sigma \in PC_S^{\sigma_0}} \sum_{i \in S} \alpha_i C_i(\sigma)$$

One of the results of the paper is that sequencing games are convex (and therefore totally balanced). Urgency indices are defined in the following way. For each job i the urgency index u_i is given by $\alpha_i p_i^{-1}$. It follows from Smith's rule that an optimal order for the jobs is to arrange them according to decreasing urgency indices. Such an optimal order can be achieved by a series of switches of neighbours. Let i and j be two neighbours with i standing in front of j. The gain obtained by switching i and j is denoted by g_{ij} and given by

$$g_{ij} := \max(0, p_i\alpha_j - p_j\alpha_i) =: (p_i\alpha_j - p_j\alpha_i)_+$$

Using the g_{ij}'s an alternative expression for v can be given. Let T be a connected coalition, i.e. a coalition which consists of only one component. Then

$$v(T) = \sum_{i,j \in T, \sigma_0(i) < \sigma_0(j)} g_{ij} \ .$$

For a non-connected coalition S the total cost savings is the sum of the cost savings in all its components. Hence,

$$v(S) = \sum_{T \in S/\sigma_0} v(T) \ .$$

Furthermore, the authors introduce a simple allocation rule for the cost savings, the Equal Gain Splitting (EGS) rule. The *EGS*-rule splits g_{ij} equally between i

and j. Formally,

$$EGS_i(v) = 1/2\left(\sum_{\sigma_0(k)<\sigma_0(i)} g_{ki} + \sum_{\sigma_0(j)>\sigma_0(i)} g_{ij}\right)$$

The authors prove that EGS-rule assigns to each sequencing game a core allocation. So, we can find a core allocation without computing the values $v(S)$ of the game.

N.B. In the sequel of this paper we will use the following notation. If σ_0 is the initial order of the jobs and i and j are jobs with $\sigma_0(i) < \sigma_0(j)$, then $[i,j]_{\sigma_0} := \{k|\sigma_0(i) \leq \sigma_0(k) \leq \sigma_0(j)\}$ and we write $[i,j]$ instead of $[i,j]_{\sigma_0}$ if σ_0 is the natural order $1 \prec 2 \prec \cdots \prec n$. For a coalition T the T-unanimity game u_T is defined by $u_T(S) = 1$ if $T \subset S$ and $u_T(S) = 0$ for all other coalitions S. It is easy to check that a sequencing game v can be expressed by $v = \sum_{\sigma_0(i)<\sigma_0(j)} g_{ij} u_{[i,j]_{\sigma_0}}$.

Example: We consider a situation where there are three jobs (customers) with processing times vector $p = (2, 3, 5)$, weights vector $\alpha = (1, 3, 6)$, due dates $d = (5, 3, 8)$ and penalties for lateness $a = (1, 4, 7)$. The original order is given by the identity permutation. The admissible rearrangements of jobs for a coalition S are those that do not switch jobs that are in different components of S, i.e. no jumping over non-members is allowed.

In we consider the weighted cost criterion, we find the following sequencing game $v(1, 2) = 3$, $v(2, 3) = 3$ and $v(1, 2, 3) = 13$ (with optimal order $3 \prec 2 \prec 1$). The coalition (1.3) has value 0 as a (profitable) switch (1, 3) is not allowed since this would involve jumping over 2 who is not in the coalition. This game has a non-empty core. A core allocation is for example (4, 5, 4).

If we consider the penalties for lateness, we get another game w with the following values: $w(1, 2) = 4$ (by a switch (1, 2) the second job is no longer late and the penalty is saved), $w(2, 3) = 7$ (job 3 is no longer late). Again $w(1, 3) = 0$ for the same reason as in the game v. Finally, $w(1, 2, 3) = 10$ (the optimal order is $2 \prec 3 \prec 1$). This game also has a non-empty core. An example of a core allocation is given by (2, 4, 4).

2 Cost Allocation in the Case of Equal Processing Times

In this section we consider one machine scheduling problems under the following conditions. The processing times p_i are *one* for all jobs $i \in N$. The initial order of the jobs is

σ_0: $1 \prec 2 \prec \cdots \prec n$

and $0 = r_1 \leq r_2 \leq \cdots \leq r_n$. Furthermore, the ready times are *integers*. The cost functions f_i: $[r_i + 1, -) \to \mathbf{R}_+$ are weakly monotonic. Let $\{t_{0k}\}_{k \in N}$ fix the semi-active time table τ_0 belonging to the initial order σ_0. In order to define a cost savings game we have to give the admissible actions for a coalition $S \subset N$.

To start with, we admit every time table $\tau = \{t_k\}_{k \in N}$ with $|t_i - t_j| \geq 1$ for all jobs $i, j \in N$, $i \neq j$ (no overlap), $t_j = t_{0j}$ for all jobs $j \in N \backslash S$ (the jobs not in S take their original position) and $t_i \geq r_i$ for all jobs $i \in S$ (the ready times are respected). Let us call such time tables *S-feasible.*

The following lemma states that there is no loss of efficiency if we only allow permutations of the jobs.

Lemma 1: If τ is a S-feasible time table, then there is a S-feasible time table τ' such that all jobs of S hold a position initially also taken by a job in S and all completion times are at most as large as under τ.

Proof: The proof goes in two steps.

I) If there is a job $i \in S$ of which the starting time can be decreased without changing the job order or violating the ready time conditions we do so. Under such an action completion times decrease. After several moves we have a time table in which no such actions are possible *for job in S*, a so-called *S*-semi-active time table. Without loss of generality we may assume that τ is *S*-semi-active. Note that in a *S*-semi-active time table the starting times of all jobs are integer (since τ_0 is semi-active, $p_i = 1$ and $r_i \in \mathbf{N}$ for all $i \in N$).

II) We assume that τ is *S*-semi-active. If τ is not permuting the time slots of S under τ_0, there is a time slot $[p, p + 1]$, $p \in N$ which is empty under τ_0 and occupied under τ. Take the first time slot of this kind. The jobs $k \in N$ starting later than p in τ_0, do so because their ready time is larger than p. (since τ_0 is semi-active and the ready times are weakly increasing). Therefore the time slot $[p, p + 1]$ is taken by a job m starting before p in τ_0. Hence, in time table τ there are fewer jobs (in S) starting before time p and at least one of the time slots before $[p, p + 1]$, initially taken by a job of S is empty under τ. Take the first time slot of this kind: $[q, q + 1]$. Then there are less jobs in S starting under τ before $q + 1$ than under τ_0 and there is a job in S which starts under τ later than time q and can start at time q. If we let it start at time q we have one job with a lower completion time and the time table is still feasible. So we proved:

As long as there is a job in S in a time slot not taken under τ_0 we can decrease the completion time of at least one job in S.

After finitely many moves we have a time table τ' of the type we looked for with no completion time larger than under τ. ●

By the Lemma we can restrict the set of admissible actions of a coalition S to

$$A(S) := \{\{t_{0\sigma(i)}\}_{i \in S} | \sigma \in \Pi_S \text{ and } t_{0\sigma(i)} \geq r_i \text{ for all } i \in S\} .$$

Then the cost saving game (N, v) is defined by

$$v(S) := \sum_{i \in S} f_i(t_{0i} + 1) - \min_{\sigma \in \Pi_S} \sum_{i \in S} f_i(t_{0\sigma(i)} + 1)$$

and we can formulate the following theorem.

Theorem 2: The cooperative game (N, v) as defined above is a permutation game and consequently totally balanced.

Proof: Let M be a real number with $M > \sum_{i \in N} f_i(t_{0i} + 1)$ and define the $N \times N$-matrix A by $A_{ij} := f_i(t_{0j} + 1)$ if $t_{0j} \geq r_i$ and $A_{ij} := M$ if $t_{0j} < r_i$. We introduce the permutation game (N, w) defined by the matrix A i.e.

$$w(S) := \sum_{i \in S} A_{ii} - \min_{\sigma \in \Pi_S} \sum_{i \in S} A_{i\sigma(i)}$$

Notice that in the game w the jobs are allowed to take positions which are not feasible but they can only do so for large cost M. Since a coalition can always decide to hold the positions they initially had, no coalition will ever take an infeasible position. This means that the infeasible actions available in the definition of w are in fact never used: $w = v$. So, the game (N, v) is a permutation game and totally balanced by the main result of Tijs et al. (1984). QED

3 Σ_0-Components Additive Games and σ_0-Pairing Games

In this section we introduce a class of cooperative games for which we can determine core allocations by an easy rule (the β-rule). Next we show how cost allocation problems for certain one-machine scheduling problems lead to games in this class. Furthermore, we prove that sequencing games as defined by Curiel et al. (1989) form a subclass of our class and that the Equal Gain Splitting rule (cf. Curiel (1989)) is a special case of the β-rule. The proofs of these theorem will appear in Curiel et al. (1993).

Let N be the player set and σ_0: $1 \prec 2 \prec 3 \prec \cdots \prec n$ be a given order of the player set N. We call a cooperative game (N, v) a *σ_0-components additive game*[1] if

(a) $v(i) = 0$ for all $i \in N$ (b) v is superadditive (c) $v(S) = \sum_{T \in S/\sigma_0} v(T)$.

In addition to the set $P(\sigma_0, i)$ of predecessors of i with respect to the permutation σ_0 defined in section 1, we define the set $F(\sigma_0, i)$ of followers of i with respect to the permutation σ_0 by $F(\sigma_0, i) := \{j \in N | \sigma_0(i) < \sigma_0(j)\}$. Let $\overline{P(\sigma_0, i)}$ be the set $P(\sigma_0, i)$ together with i and let $\overline{F(\sigma_0, i)}$ be the set $F(\sigma_0, i)$ together with i. For σ_0-components additive games we define the β-rule by

$$\beta_i(v) := 1/2(v(\overline{P(\sigma_0, i)}) - v(P(\sigma_0, i)) + v(\overline{F(\sigma_0, i)}) - v(F(\sigma_0, i))) \ .$$

Comparing the β-rule with the Shapley value we see that while the Shapley value takes the average of all the marginal vectors $\psi(v)$ of the game v, the β-rule takes the average of *two* marginal vectors, the one belonging to the given order σ_0 and the one belonging to the inverse order of σ_0. In the following theorem we show that the allocation defined by the β-rule is in the core for σ_0-components additive games.

Theorem 3: The β-rule gives a core allocation for σ_0-components additive games.

Proof: (see Curiel et al. (1993)).

Sequencing games as studied in Curiel et al. (1989) are special cases of σ_0-components additive games. Other examples of σ_0-components additive games are given by the following situation. Let us consider the one-machine scheduling problem with ready times $r_i = 0$ for all jobs $i \in N$, arbitrary processing times $p_i > 0$ and weakly monotonic cost functions f_i: $[p_i, -) \to \mathbf{R}_+$. A coalition S is only allowed to reorder the jobs inside components of S. Then

$$v(S) := \sum_{i \in S} f_i(t_{0i} + p_i) - \min_{\pi \in PC_S^{\sigma_0}} \sum_{i \in S} f_i(t_{i\pi} + p_i)$$

where $t_{i\pi}$ is the starting time of job $i \in S$ under the permutation π. Theorem 4 states that the cooperative game (N, v) is a σ_0-components additive game and hence, that the β-rule gives a core allocation.

---

[1] Recently Potters and Reijnierse (1993) investigated the properties of Γ-component additive games. These games satisfy the conditions (a) and (b) and (c') $v(S) = \sum_{T \in S/\Gamma} v(T)$ wherein Γ is a tree with node set N and S/Γ is the set of components of S in Γ.

Theorem 4: Every one-machine job scheduling problem with an additive and regular cost criterion and ready times zero gives rise to a σ_0-components additive cost saving game, and consequently, the β-rule gives a core allocation of the game.

Proof: (see Curiel et al. (1993)).

Example: Let us consider a three-job one-machine problem with the following data. The initial order is given by the identity permutation and

$$p_1 = 1, p_2 = 3, p_3 = 2, d_1 = 5, d_2 = 3, d_3 = 5, a_1 = 1, a_2 = 2 \text{ and } a_3 = 2 \ .$$

Note that in the initial order job 1 is on time and job 2 and 3 are tardy. The total penalty is $2 + 2 = 4$. Coalition (1, 2) can decrease its cost by 2 by switching the two jobs causing them both to be on time: $v(1, 2) = 2$. By switching the two jobs coalition (2, 3) can also decrease its cost by 2 while coalition (1, 2, 3) can decrease its cost by 3. The β-rule yields the allocation (1/2, 2, 1/2) which is in the core.

A cooperative game v is called a *σ_0-pairing game* if v is an element of the positive cone generated by the games $\{u_{[ij]_{\sigma_0}} | \sigma_0(i) < \sigma_0(j)\}$.

Sequencing games are σ_0-pairing games but not every σ_0-pairing game is a sequencing game.

Example: Let N consist of four players and let $v = u_{[12]} + u_{[13]} + u_{[14]} + u_{[23]} + u_{[24]} + u_{[34]}$. Then $v(1, 2) = v(2, 3) = v(3, 4) = v(1, 2, 4) = v(1, 3, 4) = 1$, $v(2, 3, 4) = v(1, 2, 3) = 3$, $v(1, 2, 3, 4) = 6$ and $v(S) = 0$ for all other $S \subset N$. It follows that for this to be a sequencing game the original order should be either $1 \prec 2 \prec 3 \prec 4$ or $4 \prec 3 \prec 2 \prec 1$. Assume that the second one is the case. From $v(1, 2) = v(2, 3) = v(3, 4)$ it follows that $g_{21} = g_{32} = g_{43} = 1$. Because $v(1, 2, 3) = 3$ we have $g_{31} = 1$, and because $v(2, 3, 4) = 3$ we have $g_{42} = 1$. Finally, $v(1, 2, 3, 4) = 6$ implies $g_{41} = 1$. However, it is impossible to find processing times p_i and weights α_i such that $g_{ij} = 1$ for all $i > j$. The same difficulty is encountered if we start with the first order.

It is easy to verify that every σ_0-pairing game is a σ_0-components additive game but not every σ_0-components additive game is a σ_0-pairing game. In fact, since the games $u_{[ij]_{\sigma_0}}$, $i < j$ are convex, every σ_0-pairing games is convex whereas a σ_0-components additive game need not be convex.

Example: Let N be the set $\{1, 2, 3\}$ and $v(1, 2, 3) = 3$, $v(1, 2) = v(2, 3) = 2$ and $v(S) = 0$ for all other coalitions. This is the game from the example directly after theorem 4. We saw that (N, v) is a σ_0-components additive game with σ_0 the identity permutation. It is not convex, since $v(1, 2) + v(2, 3) > v(1, 2, 3) + v(2)$, and hence it is not a σ_0-pairing game. Note that $v = 2u_{[12]} + 2u_{[23]} - u_{[13]}$.

The following theorem describes the β-rule, the Shapley value and the τ-value for σ_0-pairing games in terms of the coefficients g_{ij}, $i < j$. For simplicity of notation we will assume throughout the theorem that σ_0 is the identity permutation. For any other permutation the proof proceeds similarly.

Theorem 5. Let (N, v) be a σ_0-pairing game and $v = \sum_{i<j} g_{ij} u_{[ij]}$. Then

1) $\beta_i(v) = 1/2(\sum_{j<i} g_{ji} + \sum_{i<k} g_{ik})$,
2) $\phi_i(v) = \sum_{j \le i \le k, j \neq k} (k - j + 1)^{-1} g_{jk}$,
3) $\tau_i(v) = \lambda M_v(i)$ *with* $\lambda = (\sum_{j<k} (k - j + 1) g_{jk})^{-1} (\sum_{i,j} g_{ij})$.

Comment: The number g_{ij} is the profit which the players i and j can make if they cooperate with the players between i and j. The β-rule makes an equal division of the profit between i and j and the Shapley value distributes the profit equally among the player i, $i + 1, \ldots$ up to j.

Proof:

1) By definition, $\beta_i(v) = 1/2(v(\overline{P(\sigma_0, i)}) - v(P(\sigma_0, i)) + v(\overline{F(\sigma_0, i)}) - v(F(\sigma_0, i)))$
Since all these coalitions are connected we find

$$\beta_i(v) = 1/2\left(\sum_{j<k\le i} g_{jk} - \sum_{j<k<i} g_{jk} + \sum_{i\le j<k} g_{jk} - \sum_{i<j<k} g_{jk} \right)$$

After deleting common terms we find $\beta_i(v) = 1/2(\sum_{j<i} g_{ji} + \sum_{i<k} g_{ik})$.
2) By the linearlity, symmetry and dummy-player property of the Shapley value we find

$$\phi_i(v) = \sum_{j<k} g_{jk} \phi(u_{[jk]})_i = \sum_{j<k} g_{jk} |k - j + 1|^{-1} 1_{[j,k]}(i)$$

$$= \sum_{j \le i \le k, j \neq k} g_{jk} |k - j + 1|^{-1}$$

where $1_{[j,k]}(i) = 1$ if $i \in [j, k]$ and 0 otherwise.
3) A σ_0-pairing game v is convex with $v(i) = 0$ for all $i \in N$. Therefore, the τ-value is a multiple of the marginal vector $M(v)$: $\tau(v) = \lambda M(v)$ with $\lambda = (\sum_{i \in N} M_i(v))^{-1} v(N)$. Cf. Driessen and Tijs (1985). Further,

$$M_i(v) = \sum_{j<k} g_{jk} - \sum_{j<k<i} g_{jk} - \sum_{i<j<k} g_{jk} = \sum_{j \le i \le k, j \neq k} g_{jk}$$

and therefore,

$$\sum_{i \in N} M_i(v) = \sum_{j<k} (k - j + 1) g_{jk} \ .$$

This finishes the proof of the theorem 5. QED

With the expression for the β-rule given in theorem 5 it is easy to see that the *EGS*-value is a special case of the β-rule.

Summarizing, we have done the following in this section. We have introduced two generalizations of sequencing games, namely σ_0-components additive games and σ_0-pairing games. The relationship between these games is as follows: every sequencing game is a σ_0-pairing game and every σ_0-pairing game is a σ_0-components additive game. All the inclusions here are strict. Furthermore, we showed that every σ_0-components additive game is balanced and that the β-rule yields a core element. Using this result we showed that except for the weighted completion time criterion considered by Curiel et al., the penalties for tardiness criterion also gives rise to balanced games. Contrary to the sequencing games studied by Curiel et al. these games need not be convex. For σ_0-pairing games a concise description of the β-rule, the Shapley-value and the τ-value were given.

4 Final Remarks

In the sections 2 and 3 we associated with a one-machine scheduling problem a cooperative game and used this game to find a fair allocation of the minimal cost. The value of a coalition S was defined as the maximal cost savings which the coalition can reach by *S-admissible rearrangements*. In section 2 where all the processing times are equal, the set of S-admissible rearrangements of the jobs is the set of all permutations of the jobs in S, whereas in section 3 where the processing times differ, the set of S-admissible rearrangements is the set of all permutations of the jobs in S which respect the components of S (with respect to the original order). The rationale behind this is that if the processing times are equal the players not in S do not mind if jobs in S jump over them in the process of switching because each job has the same processing time anyhow. However, if the processing times are not equal, then a player not in S will object to jobs in S jumping over him since he might end up with jobs in front of him that have a longer processing time than the jobs he had in front of him in the original order. But even if the processing times are not all equal, there may be more rearrangements which do not hurt the interests of the players outside the coalition S or from which they might even benefit. Therefore we introduce in this section larger classes of S-admissible actions. The first class of S-admissible time

tables can be described as follows: A time table τ is admissible if for every $j \notin S$ $t_j = t_{0j}$ and $\sigma(j) = \sigma_0(j)$. So the jobs outside of S take their original positions and the number of predecessors of every job outside of S remains the same. Members of S are allowed to jump over non-members as long as these two conditions are satisfied. A wider class of actions can be given by $t_j \leq t_{0,j}$ for every $j \notin S$ and $\sigma(j) = \sigma_0(j)$ for every $j \notin S$. Now some jobs outside of S may profit from the action of S and some jobs in S may take also indirectly advantage from the earlier completion of some jobs outside of S. Two other extensions of the set of S-admissible rearrangements are the set of rearrangements π satisfying $t_j = t_{0,j}$ and the set of rearrangements π satisfying $t_j \leq t_{0,j}$ for all $j \notin S$. In the first two sets of rearrangements the number of predecessors of a job not in S does not change whilee in the last two rearrangements this number might change as long as it does not cause a delay in the starting time of any job not in S.

An example will clarify the distinction between these classes of actions. Let N consist of four jobs $1 \prec 2 \prec 3 \prec 4$ and let $p_1 = 5$, $p_2 = 3$, $p_3 = 2$ and $p_4 = 1$. Consider the coalition $S = \{1, 3, 4\}$. The components of S are $\{1\}$ and $\{3, 4\}$. Note that the only admissible action which respect the components is the switch $(3, 4)$ with a semi-active timetable τ given by $\tau(1) = 0$, $\tau(2) = 5$, $\tau(4) = 8$ and $\tau(3) = 9$. With our four new sets of rearrangements we get the following time tables (see figure).

In the first class of actions it is possible to go to the time table

$$\tau(4) = 0, \tau(2) = 5, \tau(1) = 8 \quad \text{and} \quad \tau(3) = 13 \ .$$

In the second class we can go to the table

$$\tau(4) = 0, \tau(2) = 1, \tau(1) = 4 \quad \text{and} \quad \tau(3) = 9 \ .$$

In the third class the following time table is admitted

$$\tau(4) = 0, \tau(3) = 1, \tau(2) = 5 \quad \text{and} \quad \tau(1) = 8 \ .$$

And finally, in the fourth class we can obtain the time table

$$\tau(4) = 0, \tau(3) = 1, \tau(2) = 3 \quad \text{and} \quad \tau(1) = 6 \ .$$

The drawback of these classes of S-admissible actions is that admissibility depends on the data of the scheduling problem. An S-feasible action in one problem may be not feasible in an other problem. Further, the actions in the second and fourth class can only be performed if some players outside S are willing to shift forwards. This means that a mild form of cooperation with players outside S is required.

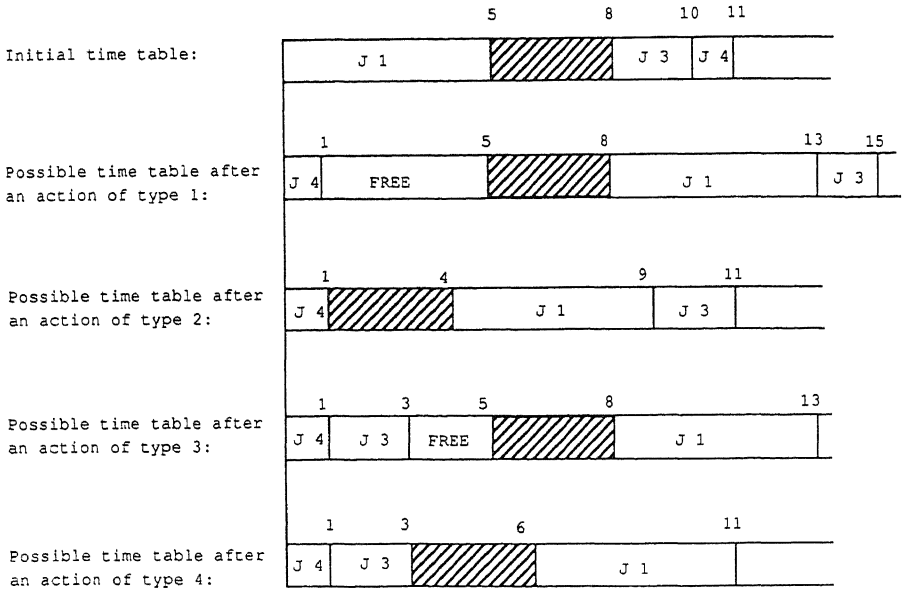

Fig. 1.

In Veltman (1988) the games arising from the second relaxation are proved to be superadditive and balanced for $n = 2$ and 3. Hamers (1988) proved by ad-hoc methods that these games are balanced for $n = 4$. Further there are no results known.

References

Curiel IJ, Pederzoli G, Tijs SH (1989) Sequencing games. European J of Operational Research 40:344–351

Curiel IJ, Potters JAM, Rajendra Prasad V, Tijs S, Veltman B (1993) Sequencing and cooperation. Forthcoming in Operations Research

Driessen TSH, Tijs SH (1985) The τ-value, the core and semiconvex games. International J of Game Theory 14:229–247

Granot D, Huberman G (1981) Minimum cost spanning tree games. Mathematical Programming 21:1–18

Hamers H (1988) Wachtrijspelen. Master's thesis Mathematics, University of Nijmegen, The Netherlands (in Dutch)

Kalai E, Zemel E (1982) Totally balanced games and games of flow. Math of Operations Research 7:476–478

Owen G (1975) On the core of linear production games. Math Programming 9:358–370

Potters JAM, Curiel IJ, Tijs SH (1992) Traveling salesman games. Mathematical Programming 53:199–211

Potters JAM, Reijnierse JH (1993) Γ-component additive games submitted to SIAM J. of discrete mathematics

Shapley LS (1953) A value for n-person games. Ann of Math Studies 28:307–317

Shapley LS (1971) The assignment game I: the core. Int Journal of Game Theory 1:111–130

Tijs SH (1981) Bounds of the core and the τ-value. In: Game theory and mathematical economics, Moeschlin O, Pallaschke D (Eds) North-Holland Publ. Cie, Amsterdam, 123–132

Tijs SH, Driessen TES (1986) Game theory and cost allocation. Management Science 32:1015–1028

Tijs SH, Parthasarathy T, Potters JAM, Rajendra Prasad V (1984) Permutation games: another class of totally balanced games. OR Spectrum 6:119–123

Veltman B (1988) Wachtrijspelen. Master's thesis Mathematics, University of Nijmegen, The Netherlands (in Dutch)